Proceedings of the 5th International Conference on Computing Innovation and Applied Physics
DOI: 10.54254/2753-8818/2026.CH31268

A Comparative Analysis of Matrix Factorization Models for Collaborative Filtering: From Basic SVD to SVD++

Feiyu Su

School of Science, Dalian Maritime University, Dalian, China
sufeiyu@dlmu.edu.cn

Abstract. Recommendation systems have permeated society and everyone's daily life with the expanding use of applications. It plays an important role in providing recommendations to active users based on either their historical interests or interests shown by similar users. This paper aims to introduce several components of a recommendation system, such as Collaborative Filtering (CF) and Matrix Factorization, and combining the dominant approach and main technique: Singular Value Decomposition (SVD). Additionally, with the advances in technology, various recently established SVD-based models are mentioned in this paper, attempting to give an answer to the question that though studies have shown the coexistence of advantages and disadvantages for each mode, the trade-offs between their predictive accuracy, computational cost, and robustness to data sparsity have still not been fully answered. Furthermore, this paper concentrates on the comparison and analysis of diverse matrix factorization variants, meanwhile evaluating their performances. For instance, basic SVD, BiasSVD, and SVD++. Finding indicates that SVD shows more efficiency in small-scale datasets, BiasSVD is suitable for cases with noticeable biases, and SVD++ gives high-quality results when handling both explicit and implicit feedback data. This study provides practical guidelines for practitioners to select the most desirable model based on their existing database.

Keywords: Recommendation Systems, Collaborative Filtering, Singular Value Decomposition, matrix factorization

1. Introduction

Nowadays, online entertainment has taken up an enormous part of everyone's life. A wide range of online platforms and applications, for instance, shopping apps and short video apps, were launched to attract customers based on a recommendation algorithm. In the process of interaction between software and users, the recommendation system assists customers by filtering products and displaying videos similar to the content in their history. A competition called "Netflix Prize" was launched by Netflix in 2006, aiming to improve its recommendation algorithm and increase the accuracy of movie recommendations. Similar perplexity appears in other corporations, hence studying and improving the implementation of the recommendation algorithm is becoming an increasingly urgent task.

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2753-8818/2026.CH31268

For traditional recommendation algorithms, there are Collaborative Filtering (CF), Content-Based Filtering (CBF), and Hybrid Recommendation. Among these, CBF is the simplest, which relies solely on features of items and products and gives proper recommendations to users. CF is being categorized into Memory-Based CF and Model-Based CF, Item-Based CF, and User-Based CF in different aspects. Nevertheless, when the number of items rises and similarity in products increases, CF is facing the challenge of the sparsity problem combined with the synonymy problem [1]. To address the challenges, Matrix Factorization (MF) is investigated, focusing on the core user-item rating matrix. Singular Value Decomposition (SVD) occupies a significant position in managing the user-item matrix. Traditional SVD algorithm and its advanced algorithms such as FunkSVD, BiasSVD, and SVD++ are already being employed in people's daily used applications. Traditional SVD, for example has application in image denoising, image compression, and image forensics [2]. SVD could manipulate the image in based on two distinctive data and noise subspaces [2]. Additionally, the DENSVD model can be seen as a further extension of BiasSVD. Based on BiasSVD, it not only incorporates implicit feedback (like SVD) but also additionally introduces 'novelty' as a regularization term, thereby taking both accuracy and novelty into account in the optimization objective [3].

While these models are well-known, there is a lack of systematic analysis that directly compares their performances, especially concerning the impact of data sparsity on their relative advantages. In this paper, comparison of traditional SVD, FunkSVD, BiasSVD, and SVD++ will be introduced to show both the advantages and disadvantages in each variant, hence to give an illustration of model choosing.

2. Methodology and implementation

2.1. Collaborative filtering

A recommendation system is essentially an information filtering system. With the information overload on the internet nowadays, CF is designed to perform information filtering. CF is currently the most commonly used algorithm, which divides customers into groups considering their similarity in interests and gives recommendations based on account of their interests. Once analogous feedback was given by a few of group members, the algorithm will automatically regard it as common feedback for all group members. However, feedback in CF can be either explicit or implicit, for instance, ratings provided by users and browsing or purchase history [1]. Also, the result occasionally suffers from a data sparsity problem combine with time complexity when increasing items. If users rate only a few items out of a massive number of items, the accuracy of the recommended items could, to some extent, be affected [4]. Last but not the least, there are synonymy problems, cold start problems and poor scalability, and a lack of interpretability [5].

2.2. Singular value decomposition

To solve the scalability problem when extreme rise in items in the recommendation system, SVD is applied in practice. SVD follows the steps of padding matrix, decompensation and dimensional reduction. Firstly, according to database, it build the user-item rating matrix (let's call it A) in a high dimension while most elements are zero. It is clear that only a few latent factors lead to the main driver. To find the latent factors, SVD factorize the matrix A into the product of three matrices P, Σ and Q^T. Suppose $A_{m\times n}(m<n)$, then $P_{m\times m}$, $\Sigma_{m\times n}$, $Q_{n\times n}$.

DOI: 10.54254/2753-8818/2026.CH31268

$$A_{m\times n} = U_{m\times m}\,\Sigma_{m\times n}{V_{n\times n}}^{T} \tag{1}$$

$$U = [u_1, u_2, \ldots, u_m], V^T = [{v_1}^T, {v_2}^T, \ldots, {v_n}^T], \Sigma = \begin{bmatrix} \lambda_1 & \cdots & 0\ 0 & \ldots & 0 \\ & \ddots & & \ddots & \\ 0 & \cdots & \lambda_m 0 & \ldots & 0 \end{bmatrix} \tag{2}$$

U is called the User-Feature Matrix, which row number is equal to user number and column number match the number of latent factors. u_i represent the degree of preference of all users for potential latent factor i. V^T is the Item-Feature Matrix, which column number match the number of latent factors when row number equals to items number. ${v_i}^T$ symbolizes the weight of all items on all the latent factor j. Both U and V^T are orthogonal matrices. Σ is the Singular Value Matrix and can be either in the shape of square or rectangle. It is a diagonal matrix with singular values. The larger the singular value, the more important the factor is in explaining the original data. For the prediction part, it has rating matrix R_hat build by formula

$$R_hat\,[i,\ j]\ =\ (U\,[i,\ :]\ *\ \Sigma)\ *\ V\ \hat{}\ T\,[:,\ j] \tag{3}$$

Meaning the predicted rating is calculated by the dot product of preference vector of user i and the feature vector of item j, considering the importance of factors.

SVD is powerful in reducing space complexity, so it performs better when handling larger databases compared to traditional CF. However, traditional SVD is rarely used directly in a recommendation system in consequence of a fatal limitation—it is only capable of solving dense matrices and relies on imputation to fill in missing ratings. While imputation can be very expensive as it significantly increases the amount of data, it is not applicable in a recommendation system [6]. To handle the problem, FunkSVD came into practice. Additionally, taking into account personal preference in scoring, a bias term was added to the algorithm, and there came BiasSVD. Furthermore, by adding implicit feedback information, SVD++ was designed.

2.3. SVD in practice

2.3.1. FunkSVD (latent factor model)

FunkSVD was originally proposed by Simon Funk in the “Netflix Prize”, aiming to give a solution to the blank field of sparse matrices. The idea is to only consider the positions of elements that have exact marks instead of the whole matrix and create a loss function (usually MSE) with the real score and the predicted score. To minimize the loss function, it has the forecast number approach the real rating number. The stochastic gradient descent algorithm (SGD) is applied in machine learning and the iteration process. The algorithm finishes when the number of iterations meets the expectation or the function convergence. FunkSVD is the foundation of advanced algorithms and perfectly handle the sparsity problem with high efficiency, combining scalability.

2.3.2. BiasSVD

BiasSVD is one important improved model based on FunkSVD, which considers the “bias term”. Variation appears in individuals, including different personal interests, grading preferences, and so on. In consequence, it introduced a biased term to quantify these indices. In fact, to make the result more precise, four variables are added to the formula, which are global bias, user bias, item bias, and

DOI: 10.54254/2753-8818/2026.CH31268

the interaction term. The learning process of BiasSVD is similar to FunkSVD, using SGD to minimize the loss function and achieve a preferable prediction of the user's interest. By adding the bias terms and interaction term, BiasSVD reach higher precision in prediction.

2.3.3. SVD++

While FunkSVD is restricted to explicit rating data, the optimized algorithm SVD++ takes into account both explicit rating data and implicit rating data. Explicit data is the concrete feedback users gave, for example, grades 1-10. In real-life scenarios, explicit data only appear on particular occasions; therefore, the introduction of implicit data seems to be a necessity. Implicit data is less visualizable. It can be a click on a certain product or a browse of an movie though unrated movie. All these can be implications of users' interests and preferences. SVD++ predicts a lot more accurate recommendations compared to FunkSVD and BiasSVD in most conditions. Also, SVD++ can be linked with various powerful models to further upgrade its performance.

3. Application

Nowadays, the underlying ideology of SVD and its derived models has integrated the architecture of modern recommendation systems. This section will introduce some derived models based on SVD.

3.1. DeepFM

DeepFM can be seen as the generalized model based on SVD, capable of solving high-order feature interactions. Also, DeepFM is a hybrid model combining matrix factorization and deep learning. The idea is based on Click-Through Rate (CTR) prediction. CTR indicates the proportion of clicks out of those who see it. Higher CTR means higher platform activity, user retention, and more outcomes. Furthermore, it is critical to use CTR to predict implicit feature interactions behind user click behaviors. Nevertheless, the problem can be very challenging and time-consuming when the input features are sparse and high-dimensional. DeepFM is an end-to-end learning model, proposed to maximize CTR for a recommendation system, meanwhile emphasizing both low- and high-order feature interactions [7].

Two components of DeepFM are the FM component and the deep component. FM is a factorization machine to learn low-order feature interactions via the dot product of their latent vectors. The Deep component is a feed-forward neural network, which is used to learn high-order feature interactions [7]. The efficiency and effectiveness of DeepFM are compared with models like FNN, IPNN. In efficiency comparison, results show DeepFM achieves the most efficiency due to the flexible architecture of the FM component, which trains features in various data records at the same time. In effectiveness comparison, the DeepFM model beats the competitors by more than 0.37% and 0.42% in terms of Area Under the ROC Curve (AUC) and Logarithmic Loss on the Company dataset [7]. This is because of the use of separate feature embeddings. The embedding layer is designed as a deep component to compress the input vector to a lower dimension to avoid the sparsity problem. It can also reduce the space complexity and time complexity [8]. In summary, DeepFM is more efficient and effective compared to other state-of-the-art models.

3.2. Neural graph collaborative filtering

Neural graph collaborative filtering (NGCF) is a generalized form of SVD++, which is exploited to explicitly encode the collaborative signals in the interaction graph structure by performing

DOI: 10.54254/2753-8818/2026.CH31268

embedding propagation. It can understand NGCF as SVD++ with a set number of high-order propagation layers, building a larger relative network with longer relation chains.

Embedding and interaction modeling are two key constituent parts of CF. In contrast to traditional models, where embeddings are directly fed into interaction layers, embeddings in NGCF are refined by being propagated on the user-item interaction graph. This consequence in users are competent to receive high-order connectivity information.

Experiments are set to show that NGCF consistently yields the best performance on all datasets compared to the strongest baselines. Also, NGCF considers multi-grained representations to infer user preference, while other models only use the output of the last layer. This demonstrates that different propagation layers encode different information in the representations [9]. Hence, as the collaborative signal can be effectively captured, NGCF might be promising in solving the sparsity issue in recommender systems [9].

3.3. RippleNet

Like DeepFM, RippleNet is also an end-to-end framework, though based on a knowledge graph (KG). A KG is like a deformation of matrix but with more interpretability and flexibility. RippleNet is designed utilizing KG for CTR prediction and knowledge-graph-aware recommendation and achieves the breakthrough from first-order statistical correlations in matrices and SVD to higher-order semantic associations in KG [10].

Uses' interests iterate along KG links to find the connected potential interests, simulating the process of seed growth. The user's interests are called a seed set, and the iteration process is called preference propagation [10]. What is special about the propagation is that multiple inputs may form a relative outcome. This is given the name ripple superposition, critical in confronting the fading relatedness.

Experiments show RippleNet outperforms baselines by 2.0% to 40.6%, 2.5% to 17.4%, and 2.6% to 22.4% on AUC in movie, book, and news recommendation, respectively [10]. It also achieves outstanding performance in top-K recommendation [10]. RippleNet was originally designed to address the sparsity problem and cold start problem of the traditional CF method. Through experiments, RippleNet indeed shows outstanding performance either in CTR or top-K recommendation [10].

4. Discussion

The traditional CF and SVD method, though it seems to have problems, for instance sparsity problem, cold start problem, and information loss in matrix decomposition, they undoubtedly establish the mindset paradigm of today's recommendation algorithms. Current improvement focuses on combining SVD with neural networks, graph theory, and deep learning. Advanced models substantially contain more variants and require more data. Take deep learning for example, it is known to be data-hungry to fully support its rich parameterization [11]. Billion-scale datasets are commonplace in applications; however, the actual situation may be one of data deficiency. Furthermore, balancing effectiveness and interpretability could be an intractable task. As more variants are brought into the formula derivation, it may not be able to give each of them a clear and detailed definition. Sometimes, it gives input to a model and receive the output from it, having no idea how it works inside it. It is like a black box full of uncertainties, which occasionally appears in neural networks and deep learning models. To handle this issue and give a specific explanation of

DOI: 10.54254/2753-8818/2026.CH31268

the recommendation-making process, either an intrinsic model whose decision mechanism is transparent is needed, or an explainer mechanism is added [12].

Fairness and bias also need to be acutely aware in recommendations. Bias usually comes from data bias and algorithm bias. Data bias lies in data generation, data collection, and data storage, while algorithm bias arises from the algorithm itself [13]. Methods to mitigate bias are briefly categorized into three parts: pre-processing method, in-processing method, and post-processing method [13]. Challenges remain, though researchers conduct. One underlying vulnerability is that the definition of fairness is not logically elucidated since the fairness demands can be contradicted from a different angle of view. To tackle the imparity, a better understanding of the core demands of individual fairness and group fairness is essential. This will lead to further investigation.

5. Conclusion

The focus of this paper is the development of CF and SVD and the comparison of different SVD-based models. This paper initially introduces CF and the basic SVD method, and additionally discusses multiple transformations based on SVD. By adding a bias term, there is BiasSVD, which achieves a more precise prediction. SVD++ is another deformation of SVD, considering both explicit data and implicit data. However, those are all in theory. To take theoretical ideas into practice while solving some potential problems in CF and SVD, this paper introduces three recently approached models: DeepFM, NGCF, and RippleNet. DeepFM reduces time complexity and space complexity, NGCF offers high-order recommendations, and RippleNet effectively addresses the sparsity problem and cold start problem. From the three example models, it seems that SVD is no longer simply an algorithm but a canonical form. Finally, the paper put forward the interpretability issue and the fairness issue, assuming possible emphasis on future research.

References

[1] Saraei, T., Benali, M., & Frayret, J.-M. (2025). A hybrid recommendation system using association rule mining, i-ALS algorithm, and SVD++ approach: A case study of a B2B company. Intelligent Systems with Applications, 25, 200477.

[2] Sadek, R. A. (2012). SVD based image processing applications: State of the art, contributions and research challenges. International Journal of Advanced Computer Science and Applications, 3(7), 26–33.

[3] Liang, X., Yu, W., & Chen, Y. (2025). Short video recommendation algorithm integrating multi-source features and novelty. Computer Applications and Software. Advance online publication.

[4] Widiyaningtyas, T., Ardiansyah, M. I., & Adji, T. B. (2022). Recommendation algorithm using SVD and weight point rank (SVD-WPR). Big Data and Cognitive Computing, 6(4), 121.

[5] Fang, W., Sha, Y., Qi, M., & Sheng, V. S. (2022). Movie recommendation algorithm based on ensemble learning. Intelligent Automation & Soft Computing, 34(1), 609–622.

[6] Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8), 30–37.

[7] Guo, H., Tang, R., Ye, Y., Li, Z., & He, X. (2020). *DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. arXiv preprint arXiv: 1703.04247

[8] Song, G., Shi, X., Xiao, Y., Wang, J., Zhang, C., & Xu, J. (2021). AutoInt: Automatic Feature Interaction Learning via Self-Attention for CTR Prediction. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining

[9] Wang, X., He, X., Wang, M., Feng, F., & Chua, T. S. (2019). Neural Graph Collaborative Filtering. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval

[10] Wang, H., Zhang, F., Xie, X., & Guo, J. (2021). RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining

DOI: 10.54254/2753-8818/2026.CH31268

[11] Zhang, S., Yao, L., Sun, A., & Tay, Y. (2018). Deep learning based recommender system: A survey and new perspectives. arXiv preprint arXiv: 1707.07435.
[12] Zhang, Y., & Chen, X. (2020). Explainable recommendation: A survey and new perspectives. Foundations and Trends® in Information Retrieval, 14(1), 1–101.
[13] Li, Y., Chen, H., Xu, S., Ge, Y., Tan, J., Liu, S., & Zhang, Y. (2022). Fairness in recommendation: A survey. arXiv preprint.